REMARKS

A. INTRODUCTION

The February 24, 2006 Office Action has been received and carefully considered. Claims 1-25 are pending in the application. In this response, no amendment has been made to the claims or other parts of the application. Applicant still believes that the application is in condition for allowance and notice thereof is respectfully requested.

B. THE REJECTION UNDER 35 U.S.C. § 103

In page 2 of the Office Action, claims 1-2, 11-13 and 25 are rejected under 35 U.S.C. §103(a) as being unpatentable over Sit *et al.* (US Patent 6,349,336, hereinafter "Sit") in view of Underwood (US Patent 6,718,535, hereinafter "Underwood"). In page 5 of the Office Action, claims 3-4 and 14-15 are rejected under 35 U.S.C. §103(a) as being unpatentable over Sit, in view of Underwood, and further in view of Fan *et al.* (US Patent 6,219,706, hereinafter "Fan"). In page 6 of the Office Action, claims 5-10 and 16-24 are rejected under 35 U.S.C. §103(a) as being unpatentable over Sit in view of Underwood and Fan, and further in view of Albert *et al.* (US Patent 6,687,222, hereinafter "Albert"). These rejections are respectfully traversed.

The obviousness rejections are improper for at least the following reasons. (1) The combinations of Sit with the other references fail to teach or suggest all the elements in the claimed invention. (2) There is no suggestion or motivation in the cited references or in the general knowledge to make the combinations.

Applicant's invention, as recited in independent claims 1 and 12, is directed to a secured file transfer protocol (FTP) system and method. Embodiments of the present invention specifically address the difficulties faced by a FTP client behind a firewall. In one embodiment, two FTP proxy systems (e.g., a FTP client proxy system 12 and a FTP server agent 14 in Figure

2) are positioned astride a firewall device. The client-side FTP proxy system (e.g., the FTP client proxy system 12) has a FTP-like session with the FTP client. The server-side FTP proxy system (e.g., the FTP server agent 14) has a FTP-like session with the FTP server. The two FTP proxy systems may communicate with each other securely across the firewall device via a single port thereon. One advantage of such an embodiment is to prevent the firewall from opening and closing random ports as in traditional FTP sessions.

Sit discloses a hypertext transfer protocol (HTTP) tunneling action that allows a remote processor to communicate with a local processor when the remote processor is coupled to the local processor via a reverse proxy device, a computer network, a firewall and a proxy agent device. The primary goal in Sit is to trick the firewall into believing that an incoming request is actually a response to an outgoing request, so that the remote processor may access/control the local processor behind the firewall. *See* Sit: col. 2, lines 39-60 and col. 3, lines 36-48.

Underwood is directed to a system and method for providing an activity framework in an e-commerce based environment. Underwood includes a voluminous description of software framework designs. However, it is believed that the Examiner is relying on Underwood only for its alleged disclosure of (1) using a single port on a firewall and (2) using proxy services for FTP.

Applicant's following arguments regarding the primary references Sit and Underwood will moot the obviousness rejections further based on Fan and Albert.

(1) The Sit-Underwood Combination Fails to Teach or Suggest All the Elements in the Claimed Invention.

As stated in MPEP § 2143.03, to establish *prima facie* obviousness of a claimed invention, all the claim limitations must be taught or suggested by the prior art. In re Royka, 490 F.2d 981, 180 USPQ 580 (C.C.P.A. 1974). That is, "[a]ll words in a claim must be considered in

judging the patentability of that claim against the prior art." In re Wilson, 424 F.2d 1382, 165 USPQ 494, 496 (C.C.P.A. 1970).

Individually or in combination, Sit and Underwood do not disclose (i) "*restricting [FTP] data flow between said first proxy system and said second proxy system to outbound communications through a single port on said firewall*" or (ii) "*restricting all flow of FTP data passing through said security system through a single port on said firewall.*" Claim 12. *See also*, Claim 1. Therefore, the Sit-Underwood combination does not render the claimed invention obvious.

Regarding Sit, a search in its disclosure reveals that Sit never uses the terms "port," "FTP" or "file transfer protocol." Indeed, in page 3 of the Office Action, the Examiner concedes that Sit does not mention using a single port on the firewall, and Sit does not mention that the system supports FTP. Since the Examiner does not allege that Sit nevertheless discloses the "single port" element through its combination with Underwood, for the obviousness rejections to stand, such element has to be found in Underwood.

To the best of Applicant's understanding, the only difference in the grounds of rejection between the current Office Action and the November 4, 2005 Final Office Action is the replacement of Epstein *et al.* (US Patent 6,584,508, hereinafter "Epstein") with Underwood. In the previous Office Action, the Examiner relied on Epstein to remedy Sit's failure to disclose an FTP service. Since Applicant does not deny the fact that the use of proxy services for FTP was well known (*see* e.g., Bellovin, Firewall-Friendly FTP, Network Working Group RFC-1579, February 1994), the only new issue arising from the replacement of Epstein with Underwood is whether Underwood indeed discloses the use of a single port on a firewall as the Examiner

alleges. As will become clear in the following discussion, Underwood does not disclose this "single port" feature as presently claimed.

The Examiner asserts that "Underwood teaches a system for providing an activity framework wherein the system funnels all traffic through a single port on the firewall instead of using a different port number for each application." Office Action, page 3. The Examiner finds support for this assertion in column 280, lines 35-38 of Underwood. This cited passage appears under a heading entitled "Circuit Proxy." The entire section of "Circuit Proxy" is quoted below with the Examiner's citation underlined.

> "Circuit Proxy
>
> A circuit proxy regulates connections between clients on the internal network and servers on the public network (and, if security policy permits, vice versa) by forcing both client and server to address their packets only to the proxy running on the firewall bastion host. These connections are established in accordance with the same types of rules as those governing packet filters and are based on the IP addresses and port numbers of client and server. <u>Unlike a packet filter, circuit proxy funnels all traffic through a single IP port (usually 1080) instead of using a different port number for each application.</u> If a client on the public network opens a session with a server on the internal network, the client has no way to learn the actual IP address of the server at the other end of the connection, since the circuit proxy intercepts all the packets.
>
> Like packet filters, circuit proxies operate at OSI layers 2 and 3 and lack complete information about a network conversation. Furthermore, circuit proxies are not transparent and may require modifications to the usage of the client and server. For this reason, circuit proxies are typically not used today."

Underwood: col. 280, lines 27-48.

It is respectfully submitted that the Examiner has misunderstood Underwood respecting its disclosure of funneling all traffic through a single IP port.

Contrary to the Examiner's assertion, Underwood's own system does not "funnel[s] all traffic through a single IP port." Rather, Underwood is referring to a well known network

protocol. Although it appears in the Detailed Description, the statement is made in the context of reviewing and evaluating firewall designs available at the time. *See*, Underwood: col. 280, lines 5-12. Indeed, Underwood states that "[a] circuit proxy," rather than the Underwood system, "funnels all traffic through a single IP port." At the end of the discussion of "Circuit Proxy," Underwood further states that "circuit proxies are typically not used today." Thus, it is apparent Underwood is not referring to a technology of his own. Applicant fully appreciates that a prior patent is taken for whatever it discloses whether or not the disclosure pertains to the patentee's own works or those of others. However, in order to fully understand Underwood's statement regarding "a single IP port," it is critical to note that Underwood is not referring to his own invention.

Underwood is actually referring to circuit proxies in general, and, although not explicitly mentioned, SOCKS Protocol in particular.

First, it is worth investigating whether a generic circuit proxy "funnels all traffic through a single IP port." The answer is negative. A "circuit proxy," also known as a "circuit-level proxy" or "circuit-level gateway," is generally defined as "a proxy service that statically defines which traffic will be forwarded." *See*, D. Newman, "Benchmarking Terminology for Firewall Performance," Network Working Group RFC-2647, August 1999, p. 6. RFC-2647 further states (emphasis added):

> The key difference between application and circuit proxies is that the latter are static and thus will always set up a connection if the DUT/SUT's rule set allows it. For example, if a firewall's rule set permits ftp connections, a circuit proxy will always forward traffic on TCP port 20 (ftp-data) even if no control connection was first established on TCP port 21 (ftp-control).

Id. Here, the discussion of a generic circuit proxy contemplates the use of different TCP ports (20 and 21) for ftp-data and ftp-control, just as recommended by the Internet Assigned Numbers

Authority (IANA) at http://www.iana.org/assignments/port-numbers. Thus, it is clear that a generic or typical circuit proxy does not necessarily restrict all its traffic to a single TCP port, at least not in FTP services. In fact, Applicant is not aware of any literature that describes a generic circuit proxy with such an attribute.

Next, it should be determined whether Underwood is referring to any special circuit proxy that has the alleged function. Note that Underwood states that "circuit proxy funnels all traffic through a single IP port (usually 1080) ..." It so happens that port 1080 is a well-known port number that IANA has assigned to SOCKS (short for "SOCKetS"), a standard for regulating network traffic that traverse a firewall. *See*, M. Leech et al., "SOCKS Protocol Version 5," Network Working Group RFC-1928, March 1996. A server implementing SOCKS Protocol is referred to as a SOCKS proxy, which is the most common type of circuit proxy. Therefore, based on the specific reference to port number 1080 in the context of circuit proxy, it is safe to conclude that Underwood is discussing a functionality of a SOCKS proxy.

Having concluded that Underwood is referring to a SOCKS proxy, it is now possible to determine what Underwood *actually* meant by the statement that "[a] circuit proxy funnels all traffic through a single IP port (usually 1080) instead of using a different port number for each application." For this determination, it is necessary to consult the SOCKS Protocol to understand how a SOCKS proxy regulates network traffic. RFC-1928 provides an authoritative description of SOCKS Protocol Version 5, which already became a *de facto* standard by the time Underwood was filed.

It is believed that Underwood's statement regarding "a single IP port (usually 1080)" originated from the following passage in RFC-1928:

> When a TCP-based client wishes to establish a connection to an
> object that is reachable only via a firewall (such determination is

> left up to the implementation), it must open a TCP connection to the appropriate SOCKS port on the SOCKS server system. The SOCKS service is conventionally located on TCP port 1080.

Id. at p. 2. This passage specifies that, for a client to connect to an object across a firewall, the client must *initially* connect to a SOCKS server (typically in the firewall) via a single port (i.e., 1080) on the SOCKS server. Thus, for all applications or clients behind a firewall to communicate with external hosts, they must direct their initial requests to the SOCKS server in the firewall. In other words, the same port number 1080 is used for the initial connections of all applications. It is in this sense that the SOCKS server "funnels *all traffic* through a single IP port (usually 1080) instead of using a different port number for each application."

However, it should be noted that this passage only applies to initial connections between the clients and the SOCKS server. This passage does not suggest that *all data* associated with the clients' subsequent communications will be restricted to this single port 1080. According to the SOCKS Protocol, once a client has established the initial connection with the SOCKS server using a CONNECT request, the SOCKS server may or may not open additional ports depending on the client's specific needs. One application that typically requires additional ports is FTP. In page 6, RFC-1928 describes a BIND request with a specific reference to FTP. The relevant passages are quoted below (emphasis added):

> BIND
>
> The BIND request is used in protocols which require the client to accept connections from the server. FTP is a well-known example, which uses the primary client-to-server connection for commands and status reports, but may use a server-to-client connection for transferring data on demand (e.g. LS, GET, PUT).
>
> It is expected that the client side of an application protocol will use the BIND request only to establish secondary connections after a primary connection is established using CONNECT. In is

> expected that a SOCKS server will use DST.ADDR and DST.PORT in evaluating the BIND request.
>
> Two replies are sent from the SOCKS server to the client during a BIND operation. The first is sent after the server creates and binds a new socket. The BND.PORT field contains the port number that the SOCKS server assigned to listen for an incoming connection. The BND.ADDR field contains the associated IP address. The client will typically use these pieces of information to notify (via the primary or control connection) the application server of the rendezvous address. The second reply occurs only after the anticipated incoming connection succeeds or fails.

According to these passages, the SOCKS server must assign a new port number (for a new socket) in addition to the 1080 port, in order to accommodate the FTP server-to-client connection (ftp-data). Therefore, at least for FTP purposes, the SOCKS server (or the firewall) must open two ports instead of one.

In view of the foregoing, it is respectfully submitted that, by "all traffic," Underwood only meant initial connections to a single port on a SOCKS server. Underwood does not disclose using a single port on a firewall to accommodate all data communications between all applications. If Underwood really meant to restrict all data to a single port,[1] the tremendous load on such single port would render such implementation impracticable, if not entirely inoperable. Although it is theoretically possible for multiple processes to share a single port, it is well known that, realistically, a typical firewall cannot and need not limit all traffic to a single port. Such a radical departure from common practice simply makes no sense.

[1] Note that Underwood does not refer to FTP in particular but speaks of "*each application*." That is, if the Examiner's understanding were correct, Underwood would be suggesting using a single port to handle *all data* from *all applications* behind a firewall.

Since neither Sit nor Underwood teaches or suggests *"restricting all flow of FTP data passing through said security system through a single port on said firewall,"* their combination cannot render the claimed invention obvious.

(2) There Is No Suggestion or Motivation to Combine or Modify Sit and Underwood.

As stated in MPEP § 2143.01, obviousness can only be established by combining or modifying the teachings of the prior art to produce the claimed invention where there is some teaching, suggestion, or motivation to do so found either in the references themselves or in the knowledge generally available to one of ordinary skill in the art. In re Fine, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988); In re Jones, 958 F.2d 347, 21 USPQ2d 1941 (Fed. Cir. 1992).

Since no such "teaching, suggestion, or motivation" can be found in the cited references or in general knowledge, the obviousness rejections of the pending claims are improper.

The text of Sit does not provide any explicit suggestion or motivation to combine with Underwood. Sit makes no reference whatsoever to the terms "FTP" or "file transfer protocol." Therefore, as the Examiner concedes in page 3 of the Office Action, Sit provides no explicit motivation to modify its system for FTP sessions.

Neither is there any implicit suggestion for the modification. First, Sit focuses exclusively on HTTP sessions, which are completely different from typical FTP sessions in terms of the required number of connections and firewall ports. It is hardly obvious how such a HTTP-specific implementation could be adapted for the kind of FTP traffic as described in the present application. Second, Sit's primary goal is to allow an outside computer to access and control a local computer behind a firewall. To achieve this goal, Sit implements two HTTP proxies to trick the firewall into believing the incoming requests are responses to some outgoing requests. This trickery on the firewall achieves exactly what a secured FTP architecture tries to

avoid. In the present invention, the security of the firewall is not in anyway circumvented or compromised. Claim 1 recites "said firewall restricting data flow between said first proxy system and said second proxy system to outbound communications through a single port on said firewall." Thus, the firewall in the present invention still functions as it is designed to. All FTP data between a local FTP client and an external FTP server are multiplexed onto a single-port secured connection between the two proxy systems. It is difficult to imagine that a network engineer mindful of firewall security would be inspired by Sit's security-bypass measures to build a secured FTP system as claimed.

Nor does Underwood provide any suggestion or motivation to combine with Sit.

First, Underwood is focused on e-Commerce which is known to be dominated by the use of Web browsers with HTML language. According to Underwood, "[a] preferred embodiment of the invention utilizes HyperText Markup Language (HTML) to implement documents on the Internet together with a general-purpose secure communication protocol for a transport medium between the client and a company." Underwood: col. 15, line 64 - col. 16, line 1. A cursory review of Underwood would reveal that its disclosure is mostly concerned with HTTP.

Second, Underwood does not even use FTP protocol for file transfer services. Underwood's system is referred to as "Resources eCommerce Technology Architecture (ReTA)." Underwood: col. 10, lines 22-23. "ReTA implements file transfer services through Microsoft's Internet Information Server 4.0 (IIS) using the HyperText Transfer Protocol (HTTP)." Underwood: col. 115, lines 37-39. Underwood apparently favors HTTP over FTP because "HTTP reduces the inefficiencies of the FTP protocol." Underwood: col. 115, line 43.

In addition, Underwood suggests "Do not mix HTTP with anonymous FTP." Underwood: col. 302, line 33.

In view of Underwood's preference for HTTP, its rejection of FTP, and its advice of not mixing these two protocols, it is inconceivable that Underwood provides any suggestion or motivation to combine with Sit's HTTP-oriented system.

Since neither Sit nor Underwood provides any motivation to combine, in order for the obviousness rejection to stand, such motivation must come from the knowledge generally available to one of ordinary skill in the art. However, that is not the case here. In order to solve the problems uniquely associated with FTP sessions through a firewall, an artisan must first identify such problems. As recognized in the present application, the specific problems include, for example, the "potential security exposures" caused by "dynamic opening and closing of ports on a firewall," and the "significant administrative resources" "required to configure a firewall to allow communication over a large range of sources and destinations." Page 9, lines 17-21. The recognition of such problems is an essential part of the present invention, which leads to a secured FTP architecture as claimed. Yet, there is no indication in the cited references that these problems were ever recognized or identified prior to the time of the present invention. Nor are these problems easily recognizable by a person of ordinary skill in the art.

Further, the HTTP-based Sit system cannot be mechanically combined with Underwood for implementation of a secured FTP system as claimed. Even if Sit and Underwood were combinable, the combination does not disclose each and every element in the claimed invention. Further, the mere fact that references can be combined or modified does not render the resultant combination obvious unless the prior art also suggests the desirability of the combination. In re Mills, 916 F.2d 680, 16 USPQ2d 1430 (Fed. Cir. 1990). Without a clear recognition of the problems associated with FTP sessions through a firewall, the desirability of the combination is not apparent from the cited references.

Since the requisite suggestion or motivation is not found in the cited references or in the knowledge generally available, the Office Action has failed to establish a *prima facie* case of obviousness. Withdrawal of the obviousness rejection is respectfully solicited.

C. CONCLUSION

For at least the reasons provided above, Applicant respectfully submits that the application is in condition for allowance. Favorable reconsideration and allowance of the pending claims are respectfully solicited.

Should there be anything further required to place the application in better condition for allowance, the Examiner is invited to contact Applicant's undersigned representative at the telephone number listed below before issuance of any further office action.

In the event any additional fees are due, the Commissioner is hereby authorized to charge the undersigned's Deposit Account No. 50-0206.

Respectfully submitted,

HUNTON & WILLIAMS, LLP

By: ____________________
Ce Li
Registration No. L0214

Hunton & Williams, LLP
1900 K Street, N.W., Suite 1200
Washington, D.C. 20006-1109
Telephone (202) 955-1500
Facsimile (202) 778-2201

Dated: April 7, 2006

Appendix A. RFC-2647

"Benchmarking Terminology for Firewall Performance"

Network Working Group
Request for Comments: 2647
Category: Informational

D. Newman
Data Communications

August 1999

Benchmarking Terminology for Firewall Performance

Status of this Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

Table of Contents

1. Introduction

This document defines terms used in measuring the performance of firewalls. It extends the terminology already used for benchmarking routers and switches with definitions specific to firewalls.

Forwarding rate and connection-oriented measurements are the primary metrics used in this document.

Why do we need firewall performance measurements? First, despite the rapid rise in firewall deployment, there is no standard method of performance measurement. Second, implementations vary widely, making it difficult to do direct performance comparisons. Finally, more and more organizations are deploying firewalls on internal networks operating at relatively high speeds, while most firewall implementations remain optimized for use over relatively low-speed wide-area connections. As a result, users are often unsure whether the products they buy will stand up to relatively heavy loads.

2. Existing definitions

This document uses the conceptual framework established in RFCs 1242 and 2544 (for routers) and RFC 2285 (for switches). The router and switch documents contain discussions of several terms relevant to benchmarking the performance of firewalls. Readers should consult the router and switch documents before making use of this document.

This document uses the definition format described in RFC 1242, Section 2. The sections in each definition are: definition, discussion, measurement units (optional), issues (optional), and cross-references.

3. Term definitions

3.1 Allowed traffic

Definition:
Packets forwarded as a result of the rule set of the device under test/system under test (DUT/SUT).

Discussion:
Firewalls typically are configured to forward only those packets explicitly permitted in the rule set. Forwarded packets must be included in calculating the bit forwarding rate or maximum bit forwarding rate of the DUT/SUT. All other packets must not be included in bit forwarding rate calculations.

This document assumes 1:1 correspondence of allowed traffic offered to the DUT/SUT and forwarded by the DUT/SUT. There are cases where the DUT/SUT may forward more traffic than it is offered; for example, the DUT/SUT may act as a mail exploder or a multicast server. Any attempt to benchmark forwarding rates of such traffic must include a description of how much traffic the tester expects to be forwarded.

Unit of measurement:
not applicable

Issues:

See also:
policy
rule set

3.2 Application proxy

Definition:
A proxy service that is set up and torn down in response to a client request, rather than existing on a static basis.

Discussion:
Circuit proxies always forward packets containing a given port number if that port number is permitted by the rule set. Application proxies, in contrast, forward packets only once a connection has been established using some known protocol. When the connection closes, a firewall using applicaton proxies rejects individual packets, even if they contain port numbers allowed by a rule set.

Unit of measurement:
not applicable

Issues:
circuit proxy
rule sets

See also:
allowed traffic
circuit proxy
proxy
rejected traffic
rule set

3.3 Authentication

Definition:
The process of verifying that a user requesting a network resource is who he, she, or it claims to be, and vice versa.

Discussion:
Trust is a critical concept in network security. Any network resource (such as a file server or printer) typically requires authentication before granting access.

Authentication takes many forms, including but not limited to IP addresses; TCP or UDP port numbers; passwords; external token authentication cards; and biometric identification such as signature, speech, or retina recognition systems.

The entity being authenticated might be the client machine (for example, by proving that a given IP source address really is that address, and not a rogue machine spoofing that address) or a user (by proving that the user really is who he, she, or it claims to be). Servers might also authenticate themselves to clients.

Testers should be aware that in an increasingly mobile society, authentication based on machine-specific criteria such as an IP address or port number is not equivalent to verifying that a given individual is making an access request. At this writing systems that verify the identity of users are typically external to the firewall, and may introduce additional latency to the overall SUT.

Unit of measurement:
not applicable

Issues:

See also:
user

3.4 Bit forwarding rate

Definition:
The number of bits per second of allowed traffic a DUT/SUT can be observed to transmit to the correct destination interface(s) in response to a specified offered load.

Discussion:
This definition differs substantially from section 3.17 of RFC 1242 and section 3.6.1 of RFC 2285.

Unlike both RFCs 1242 and 2285, this definition introduces the notion of different classes of traffic: allowed, illegal, and rejected (see definitions for each term). For benchmarking purposes, it is assumed that bit forwarding rate measurements include only allowed traffic.

Unlike RFC 1242, there is no reference to lost or retransmitted data. Forwarding rate is assumed to be a goodput measurement, in that only data successfully forwarded to the destination interface is measured. Bit forwarding rate must be measured in relation to the offered load. Bit forwarding rate may be measured with differed load levels, traffic orientation, and traffic distribution.

Unlike RFC 2285, this measurement counts bits per second rather than frames per second. Testers interested in frame (or frame-like) measurements should use units of transfer.

Unit of measurement:
bits per second

Issues:
Allowed traffic vs. rejected traffic

See also:
allowed traffic
goodput
illegal traffic
rejected traffic
unit of transfer

3.5 Circuit proxy

Definition:
A proxy service that statically defines which traffic will be forwarded.

Discussion:
The key difference between application and circuit proxies is that the latter are static and thus will always set up a connection if the DUT/SUT's rule set allows it. For example, if a firewall's rule set permits ftp connections, a circuit proxy will always forward traffic on TCP port 20 (ftp-data) even if no control connection was first established on TCP port 21 (ftp-control).

Unit of measurement:
not applicable

Issues:
application proxy
rule sets

See also:
allowed traffic
application proxy
proxy
rejected traffic
rule set

3.6 Concurrent connections

Definition:
The aggregate number of simultaneous connections between hosts across the DUT/SUT, or between hosts and the DUT/SUT.

Discussion:
The number of concurrent connections a firewall can support is just as important a metric for some users as maximum bit forwarding rate.

While "connection" describes only a state and not necessarily the transfer of data, concurrency assumes that all existing connections are in fact capable of transferring data. If a data cannot be sent over a connection, that connection should not be counted toward the number of concurrent connections.

Further, this definition assumes that the ability (or lack thereof) to transfer data on a given connection is solely the responsibility of the DUT/SUT. For example, a TCP connection that a DUT/SUT has

left in a FIN_WAIT_2 state clearly should not be counted. But another connection that has temporarily stopped transferring data because some external device has restricted the flow of data is not necessarily defunct. The tester should take measures to isolate changes in connection state to those effected by the DUT/SUT.

Unit of measurement:
Concurrent connections
Maximum number of concurrent connections

Issues:

See also:
connections
connection establishment time
connection overhead

3.7 Connection

Definition:
A state in which two hosts, or a host and the DUT/SUT, agree to exchange data using a known protocol.

Discussion:
A connection is an abstraction describing an agreement between two nodes: One agrees to send data and the other agrees to receive it.

Connections might use TCP, but they don't have to. Other protocols such as ATM also might be used, either instead of or in addition to TCP connections.

What constitutes a connection depends on the application. For a native ATM application, connections and virtual circuits may be synonymous. For TCP/IP applications on ATM networks (where multiple TCP connections may ride over a single ATM virtual circuit), the number of TCP connections may be the most important consideration.

Additionally, in some cases firewalls may handle a mixture of native TCP and native ATM connections. In this situation, the wrappers around user data will differ. The most meaningful metric describes what an end-user will see.

Data connections describe state, not data transfer. The existence of a connection does not imply that data travels on that connection at any given time, although if data cannot be forwarded on a previously established connection that connection should not be considered in any aggregrate connection count (see concurrent connections).

A firewall's architecture dictates where a connection terminates. In the case of application or circuit proxy firewalls, a connection terminates at the DUT/SUT. But firewalls using packet filtering or stateful packet filtering designs act only as passthrough devices, in that they reside between two connection endpoints. Regardless of firewall architecture, the number of data connections is still relevant, since all firewalls perform some form of connection maintenance; at the very least, all check connection requests against their rule sets.

Further, note that connection is not an atomic unit of measurement in that it does not describe the various steps involved in connection setup, maintenance, and teardown. Testers may wish to take separate measurements of each of these components.

When benchmarking firewall performance, it's important to identify the connection establishment and teardown procedures, as these must not be included when measuring steady-state forwarding rates. Further, forwarding rates must be measured only after any security associations have been established.

Though it seems paradoxical, connectionless protocols such as UDP may also involve connections, at least for the purposes of firewall performance measurement. For example, one host may send UDP packets to another across a firewall. If the destination host is listening on the correct UDP port, it receives the UDP packets. For the purposes of firewall performance measurement, this is considered a connection.

Unit of measurement:
concurrent connections
connection
connection establishment time
maximum number of concurrent connections
connection teardown time

Issues:
application proxy vs. stateful packet filtering
TCP/IP vs. ATM

connection-oriented vs. connectionless

See also:
data source
concurrent connections
connection establishment

connection establishment time
connection teardown
connection teardown time

3.8 Connection establishment

Definition:
The data exchanged between hosts, or between a host and the DUT/SUT, to initiate a connection.

Discussion:
Connection-oriented protocols like TCP have a proscribed handshaking procedure when launching a connection. When benchmarking firewall performance, it is import to identify this handshaking procedure so that it is not included in measurements of bit forwarding rate or UOTs per second.

Testers may also be interested in measurements of connection establishment time through or with a given DUT/SUT.

Unit of measurement:
not applicable

See also:
connection
connection establishement time
connection maintenance
connection teardown

Issues:
not applicable

3.9 Connection establishment time

Definition:
The length of time needed for two hosts, or a host and the DUT/SUT, to agree to set up a connection using a known protocol.

Discussion:
Each connection-oriented protocol has its own defined mechanisms for setting up a connection. For purposes of benchmarking firewall performance, this shall be the interval between receipt of the first bit of the first octet of the packet carrying a connection establishment request on a DUT/SUT interface until transmission of the last bit of the last octet of the last packet of the connection setup traffic headed in the opposite direction.

This definition applies only to connection-oriented protocols such as TCP. For connectionless protocols such as UDP, the notion of connection establishment time is not meaningful.

Unit of measurement:
Connection establishment time

Issues:

See also:
concurrent connections
connection
connection maintenance

3.10 Connection maintenance

Definition:
The data exchanged between hosts, or between a host and the DUT/SUT, to ensure a connection is kept alive.

Discussion:
Some implementations of TCP and other connection-oriented protocols use "keep-alive" data to maintain a connection during periods where no user data is exchanged.

When benchmarking firewall performance, it is useful to identfy connection maintenance traffic as distinct from UOTs per second. Given that maintenance traffic may be characterized by short bursts at periodical intervals, it may not be possible to describe a steady-state forwarding rate for maintenance traffic. One possible approach is to identify the quantity of maintenance traffic, in bytes or bits, over a given interval, and divide through to derive a measurement of maintenance traffic forwarding rate.

Unit of measurement:
maintenance traffic
forwarding rate

See also:
connection
connection establishment time
connection teardown
connection teardown time

Issues:
not applicable

3.11 Connection overhead

Definition:
The degradation in bit forwarding rate, if any, observed as a result of the addition of one connection between two hosts through the DUT/SUT, or the addition of one connection from a host to the DUT/SUT.

Discussion:
The memory cost of connection establishment and maintenance is highly implementation-specific. This metric is intended to describe that cost in a method visible outside the firewall.

It may also be desirable to invert this metric to show the performance improvement as a result of tearing down one connection.

Unit of measurement:
bit forwarding rate

Issues:

3.12 Connection teardown

Definition:
The data exchanged between hosts, or between a host and the DUT/SUT, to close a connection.

Discussion:
Connection-oriented protocols like TCP follow a stated procedure when ending a connection. When benchmarking firewall performance, it is important to identify the teardown procedure so that it is not included in measurements of bit forwarding rate or UOTs per second.

Testers may also be interested in measurements of connection teardown time through or with a given DUT/SUT.

Unit of measurement:
not applicable

See also:
connection teardown time

Issues:
not applicable

3.13 Connection teardown time

Definition:
The length of time needed for two hosts, or a host and the DUT/SUT, to agree to tear down a connection using a known protocol.

Discussion:
Each connection-oriented protocol has its own defined mechanisms for dropping a connection. For purposes of benchmarking firewall performance, this shall be the interval between receipt of the first bit of the first octet of the packet carrying a connection teardown request on a DUT/SUT interface until transmission of the last bit of the last octet of the last packet of the connection teardown traffic headed in the opposite direction.

This definition applies only to connection-oriented protocols such as TCP. For connectionless protocols such as UDP, the notion of connection teardown time is not meaningful.

Unit of measurement:
Connection teardown time

Issues:

See also:
concurrent connections
connection
connection maintenance

3.14 Data source

Definition:
A host capable of generating traffic to the DUT/SUT.

Discussion:
One data source may emulate multiple users or hosts. In addition, one data source may offer traffic to multiple network interfaces on the DUT/SUT.

The term "data source" is deliberately independent of any number of users. It is useful to think of data sources simply as traffic generators, without any correlation to any given number of users.

Unit of measurement:
not applicable

Issues:
user

See also:
connection
user

3.15 Demilitarized zone

Definition:
A network segment or segments located between protected and unprotected networks.

Discussion:
As an extra security measure, networks may be designed such that protected and unprotected segments are never directly connected. Instead, firewalls (and possibly public resources such as HTTP or FTP servers) reside on a so-called DMZ network.

DMZ networks are sometimes called perimeter networks.

Unit of measurement:
not applicable

Issues:
Homed

See also:
protected network
unprotected network

3.16 Firewall

Definition:
A device or group of devices that enforces an access control policy between networks.

Discussion:
While there are many different ways to accomplish it, all firewalls do the same thing: control access between networks.

The most common configuration involves a firewall connecting two segments (one protected and one unprotected), but this is not the only possible configuration. Many firewalls support tri-homing, allowing use of a DMZ network. It is possible for a firewall to accommodate more than three interfaces, each attached to a different network segment.

The criteria by which access are controlled are not specified here. Typically this has been done using network- or transport-layer criteria (such as IP subnet or TCP port number), but there is no

reason this must always be so. A growing number of firewalls are controlling access at the application layer, using user identification as the criterion. And firewalls for ATM networks may control access based on data link-layer criteria.

Unit of measurement:
not applicable

Issues:

See also:
DMZ
tri-homed
user

3.17 Goodput

Definition:
The number of bits per unit of time forwarded to the correct destination interface of the DUT/SUT, minus any bits lost or retransmitted.

Discussion:
Firewalls are generally insensitive to packet loss in the network. As such, measurements of gross bit forwarding rates are not meaningful since (in the case of proxy-based and stateful packet filtering firewalls) a receiving endpoint directly attached to a DUT/SUT would not receive any data dropped by the DUT/SUT.

The type of traffic lost or retransmitted is protocol-dependent. TCP and ATM, for example, request different types of retransmissions. Testers must observe retransmitted data for the protocol in use, and subtract this quantity from measurements of gross bit forwarding rate.

Unit of measurement:
bits per second

Issues:
allowed vs. rejected traffic

See also:
allowed traffic
bit forwarding rate
rejected traffic

3.18 Homed

Definition:
The number of logical interfaces a DUT/SUT contains.

Discussion:
Firewalls typically contain at least two logical interfaces. In network topologies where a DMZ is used, the firewall usually contains at least three interfaces and is said to be tri-homed. Additional interfaces would make a firewall quad-homed, quint-homed, and so on.

It is theoretically possible for a firewall to contain one physical interface and multiple logical interfaces. This configuration is discouraged for testing purposes because of the difficulty in verifying that no leakage occurs between protected and unprotected segments.

Unit of measurement:
not applicable

Issues:

See also:
tri-homed

3.19 Illegal traffic

Definition:
Packets specified for rejection in the rule set of the DUT/SUT.

Discussion:
A buggy or misconfigured firewall might forward packets even though its rule set specifies that these packets be dropped. Illegal traffic differs from rejected traffic in that it describes all traffic specified for rejection by the rule set, while rejected traffic specifies only those packets actually dropped by the DUT/SUT.

Unit of measurement:
not applicable

Issues:

See also:
accepted traffic
policy
rejected traffic
rule set

3.20 Logging

Definition:
The recording of user requests made to the firewall.

Discussion:
Firewalls typically log all requests they handle, both allowed and rejected. For many firewall designs, logging requires a significant amount of processing overhead, especially when complex rule sets are in use.

The type and amount of data logged varies by implementation. Testers may find it desirable to log equivalent data when comparing different DUT/SUTs.

Some systems allow logging to take place on systems other than the DUT/SUT.

Unit of measurement:
not applicable

Issues:
rule sets

See also:
allowed traffic
connection
rejected traffic

3.21 Network address translation

Definition:
A method of mapping one or more private, reserved IP addresses to one or more public IP addresses.

Discussion:
In the interest of conserving the IPv4 address space, RFC 1918 proposed the use of certain private (reserved) blocks of IP addresses. Connections to public networks are made by use of a device that translates one or more RFC 1918 addresses to one or more public addresses--a network address translator (NAT).

The use of private addressing also introduces a security benefit in that RFC 1918 addresses are not visible to hosts on the public Internet.

Some NAT implementations are computationally intensive, and may affect bit forwarding rate.

Unit of measurement:
not applicable

Issues:

See also:

3.22 Packet filtering

Definition:
The process of controlling access by examining packets based on the content of packet headers.

Discussion:
Packet-filtering devices forward or deny packets based on information in each packet's header, such as IP address or TCP port number. A packet-filtering firewall uses a rule set to determine which traffic should be forwarded and which should be blocked.

Unit of measurement:
not applicable

Issues:
static vs. stateful packet filtering

See also:
application proxy
circuit proxy
proxy
rule set
stateful packet filtering

3.23 Policy

Definition:
A document defining acceptable access to protected, DMZ, and unprotected networks.

Discussion:
Security policies generally do not spell out specific configurations for firewalls; rather, they set general guidelines for what is and is not acceptable network access.

The actual mechanism for controlling access is usually the rule set implemented in the DUT/SUT.

Unit of measurement:
not applicable

Issues:

See also:
rule set

3.24 Protected network

Definition:
A network segment or segments to which access is controlled by the DUT/SUT.

Discussion:
Firewalls are intended to prevent unauthorized access either to or from the protected network. Depending on the configuration specified by the policy and rule set, the DUT/SUT may allow hosts on the protected segment to act as clients for servers on either the DMZ or the unprotected network, or both.

Protected networks are often called "internal networks." That term is not used here because firewalls increasingly are deployed within an organization, where all segments are by definition internal.

Unit of measurement:

not applicable

Issues:

See also:
demilitarized zone (DMZ)
unprotected network
policy
rule set
unprotected network

3.25 Proxy

Definition:
A request for a connection made on behalf of a host.

Discussion:
Proxy-based firewalls do not allow direct connections between hosts. Instead, two connections are established: one between the client host and the DUT/SUT, and another between the DUT/SUT and server host.

As with packet-filtering firewalls, proxy-based devices use a rule set to determine which traffic should be forwarded and which should be rejected.

There are two types of proxies: application proxies and circuit proxies.

Unit of measurement:
not applicable

Issues:
application

See also:
application proxy
circuit proxy
packet filtering
stateful packet filtering

3.26 Rejected traffic

Definition:
Packets dropped as a result of the rule set of the DUT/SUT.

Discussion:
For purposes of benchmarking firewall performance, it is expected that firewalls will reject all traffic not explicitly permitted in the rule set. Dropped packets must not be included in calculating the bit forwarding rate or maximum bit forwarding rate of the DUT/SUT.

Unit of measurement:
not applicable

Issues:

See also:
allowed traffic
illegal traffic
policy
rule set

3.27 Rule set

Definition:
The collection of access control rules that determines which packets the DUT/SUT will forward and which it will reject.

Discussion:
Rule sets control access to and from the network interfaces of the

DUT/SUT. By definition, rule sets do not apply equally to all network interfaces; otherwise there would be no need for the firewall. For benchmarking purposes, a specific rule set is typically applied to each network interface in the DUT/SUT.

The tester must describe the complete contents of the rule set of each DUT/SUT.

To ensure measurements reflect only traffic forwarded by the DUT/SUT, testers are encouraged to include a rule denying all access except for those packets allowed by the rule set.

Unit of measurement:
not applicable

Issues:

See also:
allowed traffic
demilitarized zone (DMZ)
illegal traffic
policy
protected network
rejected traffic
unprotected network

3.28 Security association

Definition:
The set of security information relating to a given network connection or set of connections.

Discussion:
This definition covers the relationship between policy and connections. Security associations (SAs) are typically set up during connection establishment, and they may be reiterated or revoked during a connection.

For purposes of benchmarking firewall performance, measurements of bit forwarding rate or UOTs per second must be taken after all security associations have been established.

Unit of measurement:
not applicable

See also:
connection
connection establishment
policy
rule set

3.29 Stateful packet filtering

Definition:
The process of forwarding or rejecting traffic based on the contents of a state table maintained by a firewall.

Discussion:
Packet filtering and proxy firewalls are essentially static, in that they always forward or reject packets based on the contents of the rule set.

In contrast, devices using stateful packet filtering will only forward packets if they correspond with state information maintained by the device about each connection. For example, a stateful packet filtering device will reject a packet on port 20 (ftp-data) if no connection has been established over the ftp control port (usually port 21).

Unit of measurement:
not applicable

Issues:

See also:
applicaton proxy
packet filtering
proxy

3.30 Tri-homed

Definition:
A firewall with three network interfaces.

Discussion:
Tri-homed firewalls connect three network segments with different network addresses. Typically, these would be protected, DMZ, and unprotected segments.

A tri-homed firewall may offer some security advantages over firewalls with two interfaces. An attacker on an unprotected network may compromise hosts on the DMZ but still not reach any hosts on the protected network.

Unit of measurement:
not applicable

Issues:
Usually the differentiator between one segment and another is its IP address. However, firewalls may connect different networks of other types, such as ATM or Netware segments.

See also:
homed

3.31 Unit of transfer

Definition:
A discrete collection of bytes comprising at least one header and optional user data.

Discussion:
This metric is intended for use in describing steady-state forwarding rate of the DUT/SUT.

The unit of transfer (UOT) definition is deliberately left open to interpretation, allowing the broadest possible application. Examples of UOTs include TCP segments, IP packets, Ethernet frames, and ATM cells.

While the definition is deliberately broad, its interpretation must not be. The tester must describe what type of UOT will be offered to the DUT/SUT, and must offer these UOTs at a consistent rate. Traffic measurement must begin after all connection establishment routines complete and before any connection completion routine begins. Further, measurements must begin after any security associations (SAs) are established and before any SA is revoked.

Testers also must compare only like UOTs. It is not appropriate, for example, to compare forwarding rates by offering 1,500-byte Ethernet UOTs to one DUT/SUT and 53-byte ATM cells to another.

Unit of measurement:
Units of transfer
Units of transfer per second

Issues:

See also:
bit forwarding rate
connection

3.32 Unprotected network

Definition:
A network segment or segments to which access is not controlled by the DUT/SUT.

Discussion:
Firewalls are deployed between protected and unprotected segments. The unprotected network is not protected by the DUT/SUT.

Note that a DUT/SUT's policy may specify hosts on an unprotected network. For example, a user on a protected network may be permitted to access an FTP server on an unprotected network. But the DUT/SUT cannot control access between hosts on the unprotected network.

Unit of measurement:
not applicable

Issues:

See also:
demilitarized zone (DMZ)
policy
protected network
rule set

3.33 User

Definition:
A person or process requesting access to resources protected by the DUT/SUT.

Discussion:

"User" is a problematic term in the context of firewall performance testing, for several reasons. First, a user may in fact be a process or processes requesting services through the DUT/SUT. Second, different "user" requests may require radically different amounts of DUT/SUT resources. Third, traffic profiles vary widely from one organization to another, making it difficult to characterize the load offered by a typical user.

For these reasons, testers should not attempt to measure DUT/SUT performance in terms of users supported. Instead, testers should describe performance in terms of maximum bit forwarding rate and maximum number of connections sustained. Further, testers should use the term "data source" rather than user to describe traffic generator(s).

Unit of measurement:

not applicable

Issues:

See also:

data source

4. Security Considerations

The primary goal of this memo is to describe terms used in benchmarking firewall performance. However, readers should be aware that there is some overlap between performance and security issues. Specifically, the optimal configuration for firewall performance may not be the most secure, and vice-versa.

Further, certain forms of attack may degrade performance. One common form of denial-of-service (DoS) attack bombards a firewall with so much rejected traffic that it cannot forward allowed traffic. DoS attacks do not always involve heavy loads; by definition, DoS describes any state in which a firewall is offered rejected traffic that prohibits it from forwarding some or all allowed traffic. Even a small amount of traffic may significantly degrade firewall performance, or stop the firewall altogether. Further, the safeguards in firewalls to guard against such attacks may have a significant negative impact on performance.

Since the library of attacks is constantly expanding, no attempt is made here to define specific attacks that may affect performance. Nonetheless, any reasonable performance benchmark should take into

consideration safeguards against such attacks. Specifically, the same safeguards should be in place when comparing performance of different firewall implementations.

5. References

Bradner, S., Ed., "Benchmarking Terminology for Network Interconnection Devices", RFC 1242, July 1991.

Bradner, S. and J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", RFC 2544, March 1999.

Mandeville, R., "Benchmarking Terminology for LAN Switching Devices", RFC 2285, February 1998.

Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G. and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, February 1996.

6. Acknowledgments

The author wishes to thank the IETF Benchmarking Working Group for agreeing to review this document. Several other persons offered valuable contributions and critiques during this project: Ted Doty (Internet Security Systems), Kevin Dubray (Ironbridge Networks), Helen Holzbaur, Dale Lancaster, Robert Mandeville, Brent Melson (NSTL), Steve Platt (NSTL), Marcus Ranum (Network Flight Recorder), Greg Shannon, Christoph Schuba (Sun Microsystems), Rick Siebenaler, and Greg Smith (Check Point Software Technologies).

7. Contact Information

David Newman
Data Communications magazine
3 Park Ave.
31st Floor
New York, NY 10016
USA

Phone: 212-592-8256
Fax: 212-592-8265
EMail: dnewman@data.com

8. Full Copyright Statement

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.

Appendix B. RFC-1928

"SOCKS Protocol Version 5"

Network Working Group
Request for Comments: 1928
Category: Standards Track

M. Leech
Bell-Northern Research Ltd
M. Ganis
International Business Machines
Y. Lee
NEC Systems Laboratory
R. Kuris
Unify Corporation
D. Koblas
Independent Consultant
L. Jones
Hewlett-Packard Company

March 1996

SOCKS Protocol Version 5

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Acknowledgments

This memo describes a protocol that is an evolution of the previous version of the protocol, version 4 [1]. This new protocol stems from active discussions and prototype implementations. The key contributors are: Marcus Leech: Bell-Northern Research, David Koblas: Independent Consultant, Ying-Da Lee: NEC Systems Laboratory, LaMont Jones: Hewlett-Packard Company, Ron Kuris: Unify Corporation, Matt Ganis: International Business Machines.

1. Introduction

The use of network firewalls, systems that effectively isolate an organizations internal network structure from an exterior network, such as the INTERNET is becoming increasingly popular. These firewall systems typically act as application-layer gateways between networks, usually offering controlled TELNET, FTP, and SMTP access. With the emergence of more sophisticated application layer protocols designed to facilitate global information discovery, there exists a need to provide a general framework for these protocols to transparently and securely traverse a firewall.

There exists, also, a need for strong authentication of such traversal in as fine-grained a manner as is practical. This requirement stems from the realization that client-server relationships emerge between the networks of various organizations, and that such relationships need to be controlled and often strongly authenticated.

The protocol described here is designed to provide a framework for client-server applications in both the TCP and UDP domains to conveniently and securely use the services of a network firewall. The protocol is conceptually a "shim-layer" between the application layer and the transport layer, and as such does not provide network-layer gateway services, such as forwarding of ICMP messages.

2. Existing practice

There currently exists a protocol, SOCKS Version 4, that provides for unsecured firewall traversal for TCP-based client-server applications, including TELNET, FTP and the popular information-discovery protocols such as HTTP, WAIS and GOPHER.

This new protocol extends the SOCKS Version 4 model to include UDP, and extends the framework to include provisions for generalized strong authentication schemes, and extends the addressing scheme to encompass domain-name and V6 IP addresses.

The implementation of the SOCKS protocol typically involves the recompilation or relinking of TCP-based client applications to use the appropriate encapsulation routines in the SOCKS library.

Note:

Unless otherwise noted, the decimal numbers appearing in packet-format diagrams represent the length of the corresponding field, in octets. Where a given octet must take on a specific value, the syntax X'hh' is used to denote the value of the single octet in that field. When the word 'Variable' is used, it indicates that the corresponding field has a variable length defined either by an associated (one or two octet) length field, or by a data type field.

3. Procedure for TCP-based clients

When a TCP-based client wishes to establish a connection to an object that is reachable only via a firewall (such determination is left up to the implementation), it must open a TCP connection to the appropriate SOCKS port on the SOCKS server system. The SOCKS service is conventionally located on TCP port 1080. If the connection request succeeds, the client enters a negotiation for the

authentication method to be used, authenticates with the chosen method, then sends a relay request. The SOCKS server evaluates the request, and either establishes the appropriate connection or denies it.

Unless otherwise noted, the decimal numbers appearing in packet-format diagrams represent the length of the corresponding field, in octets. Where a given octet must take on a specific value, the syntax X'hh' is used to denote the value of the single octet in that field. When the word 'Variable' is used, it indicates that the corresponding field has a variable length defined either by an associated (one or two octet) length field, or by a data type field.

The client connects to the server, and sends a version identifier/method selection message:

```
+----+----------+----------+
|VER | NMETHODS | METHODS  |
+----+----------+----------+
| 1  |    1     | 1 to 255 |
+----+----------+----------+
```

The VER field is set to X'05' for this version of the protocol. The NMETHODS field contains the number of method identifier octets that appear in the METHODS field.

The server selects from one of the methods given in METHODS, and sends a METHOD selection message:

```
+----+--------+
|VER | METHOD |
+----+--------+
| 1  |   1    |
+----+--------+
```

If the selected METHOD is X'FF', none of the methods listed by the client are acceptable, and the client MUST close the connection.

The values currently defined for METHOD are:

- o X'00' NO AUTHENTICATION REQUIRED
- o X'01' GSSAPI
- o X'02' USERNAME/PASSWORD
- o X'03' to X'7F' IANA ASSIGNED
- o X'80' to X'FE' RESERVED FOR PRIVATE METHODS
- o X'FF' NO ACCEPTABLE METHODS

The client and server then enter a method-specific sub-negotiation.

Descriptions of the method-dependent sub-negotiations appear in separate memos.

Developers of new METHOD support for this protocol should contact IANA for a METHOD number. The ASSIGNED NUMBERS document should be referred to for a current list of METHOD numbers and their corresponding protocols.

Compliant implementations MUST support GSSAPI and SHOULD support USERNAME/PASSWORD authentication methods.

4. Requests

Once the method-dependent subnegotiation has completed, the client sends the request details. If the negotiated method includes encapsulation for purposes of integrity checking and/or confidentiality, these requests MUST be encapsulated in the method-dependent encapsulation.

The SOCKS request is formed as follows:

```
+----+-----+-------+------+----------+----------+
|VER | CMD |  RSV  | ATYP | DST.ADDR | DST.PORT |
+----+-----+-------+------+----------+----------+
| 1  |  1  | X'00' |  1   | Variable |    2     |
+----+-----+-------+------+----------+----------+
```

Where:

- o VER protocol version: X'05'
- o CMD
  - o CONNECT X'01'
  - o BIND X'02'
  - o UDP ASSOCIATE X'03'
- o RSV RESERVED
- o ATYP address type of following address
  - o IP V4 address: X'01'
  - o DOMAINNAME: X'03'
  - o IP V6 address: X'04'
- o DST.ADDR desired destination address
- o DST.PORT desired destination port in network octet order

The SOCKS server will typically evaluate the request based on source and destination addresses, and return one or more reply messages, as appropriate for the request type.

5. Addressing

In an address field (DST.ADDR, BND.ADDR), the ATYP field specifies the type of address contained within the field:

- X'01'

the address is a version-4 IP address, with a length of 4 octets

- X'03'

the address field contains a fully-qualified domain name. The first octet of the address field contains the number of octets of name that follow, there is no terminating NUL octet.

- X'04'

the address is a version-6 IP address, with a length of 16 octets.

6. Replies

The SOCKS request information is sent by the client as soon as it has established a connection to the SOCKS server, and completed the authentication negotiations. The server evaluates the request, and returns a reply formed as follows:

VER	REP	RSV	ATYP	BND.ADDR	BND.PORT
1	1	X'00'	1	Variable	2

Where:

- VER protocol version: X'05'
- REP Reply field:
  - X'00' succeeded
  - X'01' general SOCKS server failure
  - X'02' connection not allowed by ruleset
  - X'03' Network unreachable
  - X'04' Host unreachable
  - X'05' Connection refused
  - X'06' TTL expired
  - X'07' Command not supported
  - X'08' Address type not supported
  - X'09' to X'FF' unassigned
- RSV RESERVED
- ATYP address type of following address

```
            o  IP V4 address: X'01'
            o  DOMAINNAME: X'03'
            o  IP V6 address: X'04'
          o  BND.ADDR       server bound address
          o  BND.PORT       server bound port in network octet order
```

Fields marked RESERVED (RSV) must be set to X'00'.

If the chosen method includes encapsulation for purposes of authentication, integrity and/or confidentiality, the replies are encapsulated in the method-dependent encapsulation.

CONNECT

In the reply to a CONNECT, BND.PORT contains the port number that the server assigned to connect to the target host, while BND.ADDR contains the associated IP address. The supplied BND.ADDR is often different from the IP address that the client uses to reach the SOCKS server, since such servers are often multi-homed. It is expected that the SOCKS server will use DST.ADDR and DST.PORT, and the client-side source address and port in evaluating the CONNECT request.

BIND

The BIND request is used in protocols which require the client to accept connections from the server. FTP is a well-known example, which uses the primary client-to-server connection for commands and status reports, but may use a server-to-client connection for transferring data on demand (e.g. LS, GET, PUT).

It is expected that the client side of an application protocol will use the BIND request only to establish secondary connections after a primary connection is established using CONNECT. In is expected that a SOCKS server will use DST.ADDR and DST.PORT in evaluating the BIND request.

Two replies are sent from the SOCKS server to the client during a BIND operation. The first is sent after the server creates and binds a new socket. The BND.PORT field contains the port number that the SOCKS server assigned to listen for an incoming connection. The BND.ADDR field contains the associated IP address. The client will typically use these pieces of information to notify (via the primary or control connection) the application server of the rendezvous address. The second reply occurs only after the anticipated incoming connection succeeds or fails.

In the second reply, the BND.PORT and BND.ADDR fields contain the address and port number of the connecting host.

UDP ASSOCIATE

The UDP ASSOCIATE request is used to establish an association within the UDP relay process to handle UDP datagrams. The DST.ADDR and DST.PORT fields contain the address and port that the client expects to use to send UDP datagrams on for the association. The server MAY use this information to limit access to the association. If the client is not in possesion of the information at the time of the UDP ASSOCIATE, the client MUST use a port number and address of all zeros.

A UDP association terminates when the TCP connection that the UDP ASSOCIATE request arrived on terminates.

In the reply to a UDP ASSOCIATE request, the BND.PORT and BND.ADDR fields indicate the port number/address where the client MUST send UDP request messages to be relayed.

Reply Processing

When a reply (REP value other than X'00') indicates a failure, the SOCKS server MUST terminate the TCP connection shortly after sending the reply. This must be no more than 10 seconds after detecting the condition that caused a failure.

If the reply code (REP value of X'00') indicates a success, and the request was either a BIND or a CONNECT, the client may now start passing data. If the selected authentication method supports encapsulation for the purposes of integrity, authentication and/or confidentiality, the data are encapsulated using the method-dependent encapsulation. Similarly, when data arrives at the SOCKS server for the client, the server MUST encapsulate the data as appropriate for the authentication method in use.

7. Procedure for UDP-based clients

A UDP-based client MUST send its datagrams to the UDP relay server at the UDP port indicated by BND.PORT in the reply to the UDP ASSOCIATE request. If the selected authentication method provides encapsulation for the purposes of authenticity, integrity, and/or confidentiality, the datagram MUST be encapsulated using the appropriate encapsulation. Each UDP datagram carries a UDP request header with it:

```
+----+------+------+----------+----------+----------+
|RSV | FRAG | ATYP | DST.ADDR | DST.PORT |   DATA   |
+----+------+------+----------+----------+----------+
| 2  |  1   |  1   | Variable |    2     | Variable |
+----+------+------+----------+----------+----------+
```

The fields in the UDP request header are:

- o RSV Reserved X'0000'
- o FRAG Current fragment number
- o ATYP address type of following addresses:
  - o IP V4 address: X'01'
  - o DOMAINNAME: X'03'
  - o IP V6 address: X'04'
- o DST.ADDR desired destination address
- o DST.PORT desired destination port
- o DATA user data

When a UDP relay server decides to relay a UDP datagram, it does so silently, without any notification to the requesting client. Similarly, it will drop datagrams it cannot or will not relay. When a UDP relay server receives a reply datagram from a remote host, it MUST encapsulate that datagram using the above UDP request header, and any authentication-method-dependent encapsulation.

The UDP relay server MUST acquire from the SOCKS server the expected IP address of the client that will send datagrams to the BND.PORT given in the reply to UDP ASSOCIATE. It MUST drop any datagrams arriving from any source IP address other than the one recorded for the particular association.

The FRAG field indicates whether or not this datagram is one of a number of fragments. If implemented, the high-order bit indicates end-of-fragment sequence, while a value of X'00' indicates that this datagram is standalone. Values between 1 and 127 indicate the fragment position within a fragment sequence. Each receiver will have a REASSEMBLY QUEUE and a REASSEMBLY TIMER associated with these fragments. The reassembly queue must be reinitialized and the associated fragments abandoned whenever the REASSEMBLY TIMER expires, or a new datagram arrives carrying a FRAG field whose value is less than the highest FRAG value processed for this fragment sequence. The reassembly timer MUST be no less than 5 seconds. It is recommended that fragmentation be avoided by applications wherever possible.

Implementation of fragmentation is optional; an implementation that does not support fragmentation MUST drop any datagram whose FRAG field is other than X'00'.

The programming interface for a SOCKS-aware UDP MUST report an available buffer space for UDP datagrams that is smaller than the actual space provided by the operating system:

- o if ATYP is X'01' - 10+method_dependent octets smaller
- o if ATYP is X'03' - 262+method_dependent octets smaller
- o if ATYP is X'04' - 20+method_dependent octets smaller

8. Security Considerations

This document describes a protocol for the application-layer traversal of IP network firewalls. The security of such traversal is highly dependent on the particular authentication and encapsulation methods provided in a particular implementation, and selected during negotiation between SOCKS client and SOCKS server.

Careful consideration should be given by the administrator to the selection of authentication methods.

9. References

[1] Koblas, D., "SOCKS", Proceedings: 1992 Usenix Security Symposium.

Author's Address

Marcus Leech
Bell-Northern Research Ltd
P.O. Box 3511, Stn. C,
Ottawa, ON
CANADA K1Y 4H7

Phone: (613) 763-9145
EMail: mleech@bnr.ca